Smalltalk/v สำหรับ การเรียนรู้OOP

จเร เลิศสุดวิชัย

หลังจากอ่านความรู้เบื้องต้นเรื่องของ OOP ในฉบับนี้มาบ้างแล้ว ท่านผู้อ่านหลายท่านคงคิดจะหาซอฟต์แวร์ที่ใช้หลักการของ OOP มาทดลองใช้เพื่อเสริมความเข้าใจเรื่อง OOP ให้กระจ่างชัดขึ้น ซึ่งก็มีอยู่ด้วยกันหลายตัว เช่น Smalltalk, C++, Turbo Pascal เวอร์ชัน 5.5, Actor ฯลฯ ในบทความนี้จะแนะนำซอฟต์แวร์ตัวหนึ่งซึ่งพอหาซื้อได้ในเมืองไทย คือ Smalltalk/V

ความเป็นมา

Smalltalk นั้นมีต้นกำเนิดมาจากศูนย์วิจัย Xerox Polo Alto (Xerox Polo Alto Research Center) โดยเริ่มทำการวิจัยมากว่า 10 ปีแล้ว ซึ่งในระหว่าง 10 ปีที่ทำการวิจัย ทางศูนย์วิจัยได้ออกแบบ สร้าง และนำซอฟต์แวร์ที่พัฒนาขึ้นทดลองกับผู้ใช้เป็นจำนวนมาก ซึ่งซอฟต์แวร์หรือระบบที่ออกแบบขึ้นนั้นจะตั้งชื่อตามปี คศ. ที่ได้รับการออกแบบ เช่น Smalltalk-72, Smalltalk-76 และล่าสุดคือ Smalltalk-80 งานวิจัยของศูนย์วิจัยนี้เน้นไปที่การเพิ่มความสามารถของระบบคอมพิวเตอร์ที่สามารถตอบสนองการใช้งานของผู้ใช้คอมพิวเตอร์ที่ไม่ใช่ผู้ชำนาญในการใช้งานคอมพิวเตอร์ ดังนั้นซอฟต์แวร์ที่ออกมาจึงมีลักษณะ มีความน่าใช้และสะดวกต่อการใช้งาน คืออยู่ในการทำงานลักษณะของกราฟิก มีระบบหน้าต่าง (windows), ป๊อบอัพเมนู (POP-UP menu) และสามารถจะเชื่อมต่อกับอุปกรณ์อำนวยความสะดวกเช่นเมาส์ได้

ต่อมาในปี 1979 และ 1980 บริษัท Xerox ได้ยอมอนุญาตให้เผยแพร่ Small-Talk ออกสู่วงการคอมพิวเตอร์ จุดประสงค์หลักก็เพื่อที่ขยายและเผยแพร่แนวความคิดในเรื่องของ OOP รวมทั้งหามาตรฐานการโปรแกรมแบบออปเจ็คโอเรียนเต็ดและมาตรฐานสำหรับตัว Smalltalk เอง จนกระทั่งถึงปัจจุบันก็ได้มีการพัฒนา Smalltalk-80 บนเครื่องระดับเวอร์กสเตชัน เช่น SUN, Apollo เป็นต้น ส่วน Smalltalk/V นั้นก็เป็นส่วนหนึ่งของ Smalltalk-80 ที่ทำงานอยู่บนเครื่องพีซี ความสามารถบางอย่างใน Smalltalk-80 จะไม่มีบน Smalltalk/V แต่อย่างไรก็ตาม Smalltalk/V ก็สามารถเสริมสร้างความรู้และความเข้าใจในเรื่องของ OOP ได้อย่างเพียงพอ

Smalltalk/V

Smalltalk/V ที่ใช้แนะนำในบทความนี้ เป็นเวอร์ชันหรือรุ่น 2.0 ซึ่งตัวซอฟต์แวร์ประกอบด้วย 3 แผ่น คือ แผ่นซอร์สโค้ดสำหรับระบบ แผ่น Image และแผ่น Tutorial พร้อมหนังสือคู่มือการใช้งานหนา 514 หน้า ภายในหนังสือคู่มือ ซึ่งแบ่งออกเป็นส่วนแยกกันคือ ส่วนของ tutorial อธิบายถึงระบบของ Smalltalk/V และอธิบายให้ความรู้พร้อมทั้งตัวอย่างในเรื่องของ OOP เช่น object, class ฯลฯ และส่วนที่ 2 เป็น Smalltalk/V Reference อธิบายถึงรูปแบบการใช้งาน ภาษา object, class ฯลฯ และส่วนสุดท้ายคือ Encyclopedia of Classes ซึ่งอธิบายถึง class และ method ทั้งหมดที่มีใช้ใน Smalltalk/V หนังสือคู่มือจะเป็นส่วนสำคัญที่นอกจากบอกถึงวิธีการใช้งานแล้วยังอธิบายความเข้าใจเบื้องต้นในเรื่องของ OOP ด้วย

การติดตั้ง Smalltalk/V ทำได้ง่ายมาก โดยเครื่องที่จะใช้ Smalltalk/V นี้ต้องเป็นเครื่อง IBM PC, PC/XT, PC/AT หรือเครื่องคอมแพตทิเบิ้ล โดยมีคุณสมบัติดังนี้

- มีหน่วยความจำ 512 กิโลไบต์ขึ้นไป
- มีดิสค์ไดรฟ์ 2 ตัว หรือ ฮาร์ดดิสค์และดิสค์ไดรฟ์ 1 ตัว
- จอกราฟิก CGA, EGA, Hercules, AT&T หรือ IBM
- ใช้ดอส PC-DOS หรือ MS-DOS เวอร์ชัน 2.0 ขึ้นไป

และสิ่งที่ควรจะมีเพื่อเพิ่มประสิทธิภาพการใช้งานของ Smalltalk/V คือ

- หน่วยความจำควรมีถึง 640 กิโลไบต์
- มีเมาส์ ซึ่งต้องเป็นมาตรฐานของไมโครซอฟต์ (Microsoft Compatible)

- มีโปรเซสเซอร์ช่วยคำนวณทางคณิตศาสตร์ (Floating point co-pocessor)

การติดตั้ง Smalltalk/V ถ้ามีเมาส์อยู่ด้วยจะช่วยเพิ่มความสะดวกต่อการใช้งานเป็นอย่างมาก โดยสามารถจะควบคุมเคอร์เซอร์และการเลือกหัวข้อเมนูได้รวดเร็ว แต่ถ้าไม่มีเมาส์จะต้องใช้ปุ่มลูกศรในการควบคุมแทน ซึ่งจะทำให้การทำงานช้าลงไป

และปัญหาอันหนึ่งในการติดตั้ง Smalltalk/V บนเครื่องที่มีดิสค์ไดรฟ์สองตัวนั้น นอกจากการอ่านหรือเขียนข้อมูลลงดิสค์จะช้าแล้ว ตัว Smalltalk/V ในการทำงานนั้นต้องการไฟล์ 4 ไฟล์ คือ Interpreter (V.EXE และ V2NDPART.EXE), image (IMAGE), Sources (SOURCES.SML) และ change (CHANGE.LOG) โดยที่ไฟล์ V.EXE (ขนาด 7 กิโลไบต์) และ V2NDPART.EXE (ขนาด 34 กิโลไบต์) จะต้องอยู่บนดิสค์ในโหลด Smalltalk/V ครั้งแรก ส่วนไฟล์ IMAGE นั้นจะเป็นไฟล์ที่เก็บสภาวะแวดล้อมของระบบ เช่น การเปิดปิดหน้าต่าง หรือภาวะการทำที่ค้างอยู่ก่อนเลิกใช้งาน Smalltalk/V ซึ่งขนาดไฟล์เริ่มต้นนั้นเป็น 260 กิโลไบต์และยังจะต้องขยายใหญ่ขึ้นเรื่อย ๆ แต่ใช้งานเพียงแค่ตอนเริ่มต้นและหลังการใช้งาน Smalltalk/V เท่านั้น ไฟล์ SOURCES.SML จะมีซอร์สโค้ดสำหรับ class ทั้งหมดใน Smalltalk/V (ซึ่งได้รับการ compressed โดยอ่านได้เฉพาะ Smalltalk/V เท่านั้น) ซึ่งขนาดไฟล์เริ่มต้นมีขนาด 315K และจะขยายใหญ่ขึ้นเรื่อย ๆ เมื่อมีการใช้งาน class browser ไฟล์ CHANGE.LOG มีขนาดไฟล์ 90 กิโลไบต์ ซึ่งเป็นที่เก็บซอร์สโค้ดของผู้ใช้งานเมื่อทำการแก้ไข class ที่อยู่ในระบบหรือสร้าง class ขึ้นมาใหม่ จะเห็นว่าไฟล์ IMAGE, SOURCES.SML และ CHANGE-.LOG ทั้งสามไฟล์สามารถขยายขนาดขึ้นได้เรื่อย ๆ จนบางครั้งขนาดของไฟล์แต่ละไฟล์อาจจะโตเกินที่ใส่บนแผ่นเดียวได้

การแก้ปัญหาดังกล่าวนั้น คือการป้องกันขนาดไฟล์เกินแผ่นดิสค์ สำหรับไฟล์ IMAGE นั้นสามารถลดขนาดไฟล์ลงได้ โดยก่อนจะออกหรือเลิกใช้งาน Smalltalk ควรจะปิดหน้าต่างทั้งหมดบนจอภาพก่อนจะเก็บ (save) ภาพ (Image) ส่วนไฟล์ CHANGE.-LOG ก็สามารถลดขนาดได้โดยการสร้างซอร์สไฟล์ใหม่แทน และสำหรับไฟล์ SOURCES.SML นั้นก็ทำเช่นเดียวกับไฟล์ CHANGE.LOG ซึ่งรายละเอียดเกี่ยวกับเรื่องนี้ในหนังสือคู่มือมีอธิบายไว้อย่างละเอียด

การติดตั้ง Smalltalk/V นั้นจะต้องระบุว่าจะติดตั้งบนฮาร์ดดิสค์หรือบนฟลอปปี้ดิสค์ และบอกถึงชนิดจอภาพกราฟิกที่ใช้แสดงผลด้วย ซึ่งถ้าติดตั้งบนฮาร์ดดิสค์ทำโดยการก๊อบปี้ไฟล์ทั้งหมดบนแผ่นดิสค์ทั้งสามที่ได้มาลงในฮาร์ดดิสค์ที่สร้างไดเรคทอรีขึ้นมาก่อนแล้ว จึงเรียกโปรแกรม Install ซึ่งต้องตามด้วยคำว่า hard สำหรับฮาร์ดดิสค์ หรือถ้าเป็นฟลอปปี้ดิสค์ใช้คำว่า soft แทน จากนั้นจึงตามด้วยชนิดของจอกราฟิกที่ใช้งาน การติดตั้งถ้าติดตั้งผิดสามารถทำใหม่ได้อีกโดยเรียกใช้โปรแกรม Install อีกครั้ง

การใช้งาน Smalltalk/V

ก่อนที่ทำการใช้ Smalltalk/V โดยการเขียนโปรแกรม ควรที่ทำความรู้จักกับรูปแบบของภาษา Smalltalk ซึ่งถ้าผู้อ่านที่อ่านบทความในเรื่องของ OOP ในฉบับนี้มาก่อนจะเข้าใจเร็วขึ้น โปรแกรมตัวอย่างที่จะเขียนเป็นโปรแกรมง่าย ๆ เพื่อแนะนำให้ผู้อ่านเกิดความเข้าใจในการเขียนโปรแกรมเสียก่อน รูปแบบภาษาที่กล่าวถึงจะเปรียบกับปาสคาลเป็นหลัก โดยคอลัมน์ทางซ้ายมือเป็นรูปแบบของปาสคาล และคอลัมน์ทางขวามือเป็นของ Smalltalk

การกำหนดค่าให้ตัวแปร

```
a : = b+c      a : = b+c
```

รูปแบบคำสั่งการกำหนดให้ตัวแปรใน Smalltalk จะเหมือนกับในปาสคาล ต่างกันในเรื่องการกระทำ คือ ในปาสคาลจะเป็นการนำค่า b และ c บวกกัน จากนั้นนำค่าที่ได้มาเก็บใน a แต่ใน Smalltalk นั้นการกำหนดจะเก็บพอยน์เตอร์ของออปเจ็คที่มีค่าแทน

การแบ่งแยกคำสั่ง (Statement Separator)

```
x : = 0 ;          x : = 0.
y : = 'answer' ;   y : = 'answer' .
z : = w            z : = w
```

ในปาสคาลตัวอักษรที่ใช้แบ่งแยกคำสั่งคือเครื่องหมาย " ; " แต่ใน Smalltalk จะใช้จุด " . " เป็นตัวแบ่งแยกคำสั่งแทน และก็เช่นเดียว คำสั่งสุดท้ายไม่ต้องมีตัวแบ่งแยกคำสั่ง

การเรียกฟังก์ชัน โดยใช้พารามิเตอร์

```
a : = size (array)      a : = array size
```

ในปาสคาลการเรียกฟังก์ชันจะต้องส่งค่าพารามิเตอร์ผ่านให้และส่งค่าที่คำนวณได้กลับมาเก็บในตัวแปร แต่ใน smalltalk นั้น จะเป็นการส่งผ่าน message แทน

message สามารถจะแบ่งออกเป็น 3 ส่วนคือ ตัวรับ (receiver), ตัวเลือก message (message selector) และสุดท้ายคือ พารามิเตอร์ (arguments) ซึ่งส่วน

สุดท้ายนั้นจะมีหรือไม่ก็ได้ ตัวรับและพารามิเตอร์ของ massage นั้นอาจจะเป็น expression ของ message อื่น ๆ ก็ได้

ในตัวอย่างข้างต้นนั้นไม่มีส่วนของพารามิเตอร์ ซึ่งจะเรียกว่า unary message การคำนวณ unary message จะทำจากซ้ายไปขวา เช่น

```
7 factorial sqrt
```

จะให้ผลลัพธ์ของ $\sqrt{7!}$

ต่อไปในกรณี message มีพารามิเตอร์หนึ่งตัว โดยตัว message ส่วนใหญ่จะเป็นตัวกระทำทางคณิตศาสตร์ ซึ่งเรียก message ชนิดนี้ว่า binary message ตัวอย่างเช่น

```
7 + 4
```

ซึ่งประโยคนี้จะถูกแปลว่า ส่ง message " + " และพารามิเตอร์ 4 ไปให้ออปเจ็ค 7 binary message นี้ก็เช่นเดียวกับ unary message คือกระทำเริ่มจากซ้ายไปขวา

ลักษณะของ message สุดท้ายคือ keyword message ตัวเลือก (Selector) จะประกอบด้วย keyword มากกว่าหนึ่ง keyword โดยแต่ละ keyword จะตามด้วยพารามิเตอร์ (argument) keyword ก็คือ identifier (เริ่มต้นด้วยตัวอักษรตัวเล็กตัวแรก) ซึ่งปิดท้าย keyword ด้วยเครื่องหมาย " : " ส่วนพารามิเตอร์นั้นอาจจะเป็น expression ก็ได้ ซึ่งกรณีเป็น expression ควรจะใส่ในวงเล็บเพื่อความกำกวม ตัวอย่างเช่น

```
7 max: 14
7 between: 2 and: 24
```

keyword message นั้นมีลำดับการคำนวณต่ำกว่า binary หรือ unary message เช่น

```
7 between: 2 sqrt and: 4+2
```

จะได้ค่าเหมือนกับ

```
7 between: (2 sqrt) and: (4+2)
```

การส่ง message หลาย ๆ message ให้ตัวรับเพียงตัวเดียวสามารถทำได้โดยการกั้นระหว่าง message ด้วย " ; " เช่น

```
Turtle
    white:
    home:
    go:   100;
    turn: 120;
    go:   100;
    turn: 120
```

ตัวรับสามารถจะเขียนเพียงครั้งเดียวได้ แทนการเขียนตัวรับกับ message ทั้งหมด ทำให้โปรแกรมอ่านได้ง่ายขึ้น

ประโยคเงื่อนไข (if)

```
if a < b then                    a<b
    a:= a+1;                       ifTrue [a:= a+1]
if atEnd (Stream)then            stream atEnd
    reset (Stream)                 ifTrue: [Stream reset]
else                               ifElse: [C:= Stream next]
    C:= next (Stream)
```

ลักษณะประโยคเงื่อนไขในปาสคาลและ Smalltalk จะคล้ายคลึงกัน แต่ประโยคการกระทำหลังตรวจสอบเงื่อนไขของ Smalltalk จะอยู่ในวงเล็บ "[ ]" ซึ่งเรียกว่า Block ภายใน Block อาจจะประกอบด้วยหลายประโยคก็ได้แต่กั้นด้วย " . "

การทำงานวนรอบ (Loop)

```
while i < 10 do begin   [i<10]
    sum:= sum+i             whileTrue: [
end:                        sum:= sum + 1].
```

ลักษณะการทำงานวนรอบในปาสคาลและ Smalltalk ก็จะคล้ายคลึงกัน

เอาละครับเราคงเรียนรู้กันเพียงเท่านี้ก่อน ต่อไปจะทำการเขียนโปรแกรมง่าย ๆ ใน Smalltalk/V เริ่มต้นโดยเรียก Smalltalk/V

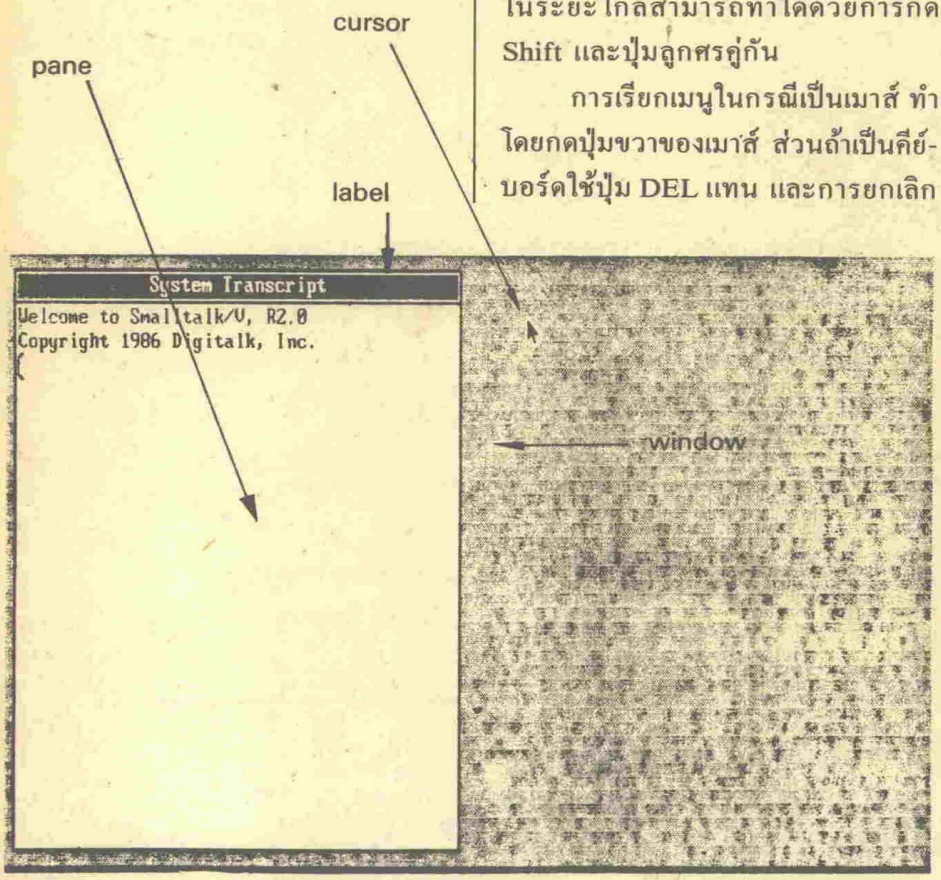
C: /smalltalk > V

Smalltalk/V จะโหลดโปรแกรมและปรากฏดังรูปที่ 1

การควบคุมทำได้ด้วยเมาส์ หรือถ้าไม่มีเมาส์ใช้ปุ่มลูกศรในการเลื่อนเคอร์เซอร์ และต้องการเลื่อนเคอร์เซอร์ในระยะไกลสามารถทำได้ด้วยการกด Shift และปุ่มลูกศรคู่กัน

การเรียกเมนูในกรณีเป็นเมาส์ ทำโดยกดปุ่มขวาของเมาส์ ส่วนถ้าเป็นคีย์บอร์ดใช้ปุ่ม DEL แทน และการยกเลิกเมนูทำโดยเลื่อนเคอร์เซอร์ออกจากเมนู ส่วนการเลือกเมนูทำโดยกดปุ่มซ้ายของเมาส์ หรือถ้าเป็นคีย์บอร์ดใช้กดปุ่ม " + " แทน

ในระบบของ Smalltalk/V นั้นจะมีเมนูอยู่ด้วยกัน 3 ชนิดคือ



รูปที่ 1 หน้าตาของ Smalltalk/V ตอนเริ่มต้น

System Menu

เป็นเมนูที่อนุญาตให้ทำเปิดหน้าต่างอื่นเพื่อทำงานอื่นของระบบ การเลิกระบบ หรือการวาดจอภาพใหม่ (redrawing screen) เป็นต้น เมนูส่วนนี้เรียกขึ้นมาได้โดยการเลื่อนเคอร์เซอร์ให้อยู่ในส่วนของนอกหน้าต่างทั้งหมดแล้วกด DEL

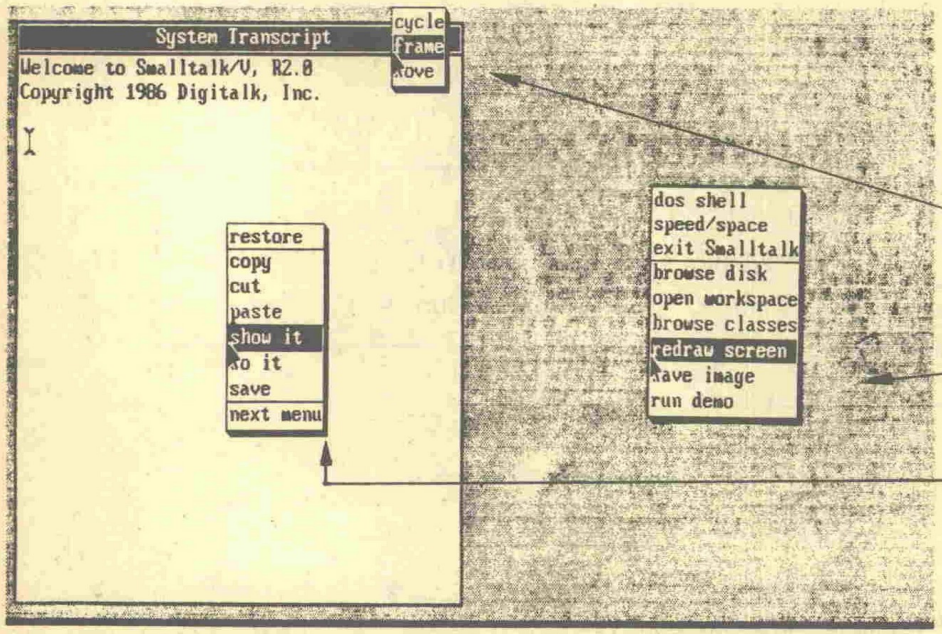
Window Menu

เป็นเมนูในการควบคุมหน้าต่าง เป็นการเลื่อนหน้าต่าง ขยายขนาด เป็นต้น เมนูส่วนนี้สามารถเรียกโดยเลื่อนเคอร์เซอร์ไปอยู่ส่วนบนของหน้าต่างที่เรียกว่า Label แล้วกด DEL

Pane Menu

เป็นเมนูที่กระทำต่อสิ่งที่อยู่ในหน้าต่าง ตามชนิดของหน้าต่างนั้น ๆ

ในหน้าต่างของ System Transcript นั้นเป็นหน้าต่างสำหรับใช้งานการแสดงข้อความโต้ตอบสำหรับการทำงานของโปรแกรม เราจะเริ่มต้นกันที่หน้าต่างนี้ บนหน้าต่างนี้จะเห็นเครื่องหมาย " (" แสดงถึงตำแหน่งของข้อความที่จะพิมพ์ลงไป เราสามารถควบคุมตำแหน่งได้โดยการเลื่อนเคอร์เซอร์ไปยังตำแหน่งที่ต้องการบนหน้าต่าง

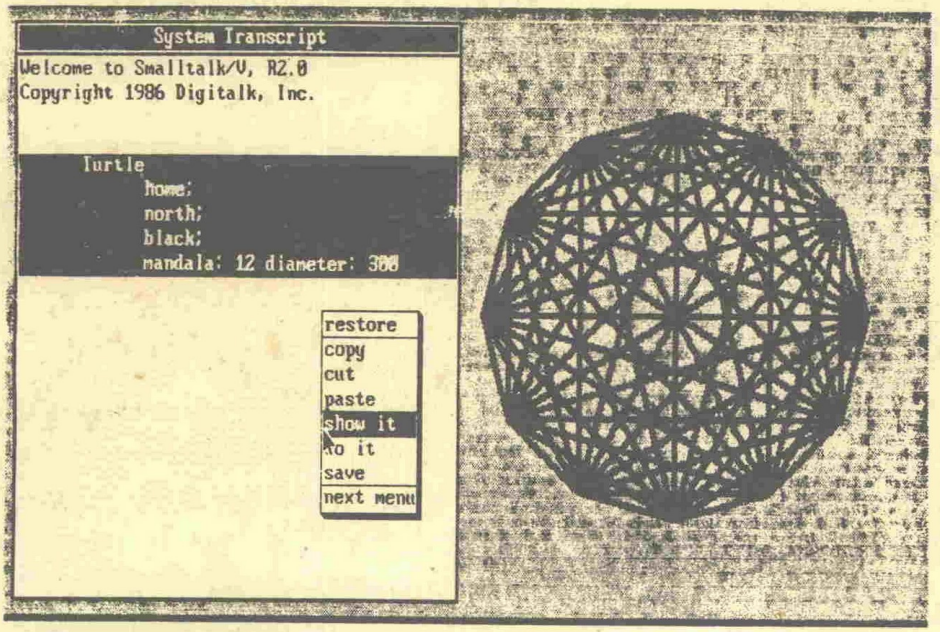


รูปที่ 2 แสดง System Menu, Window Menu และ Pane menu

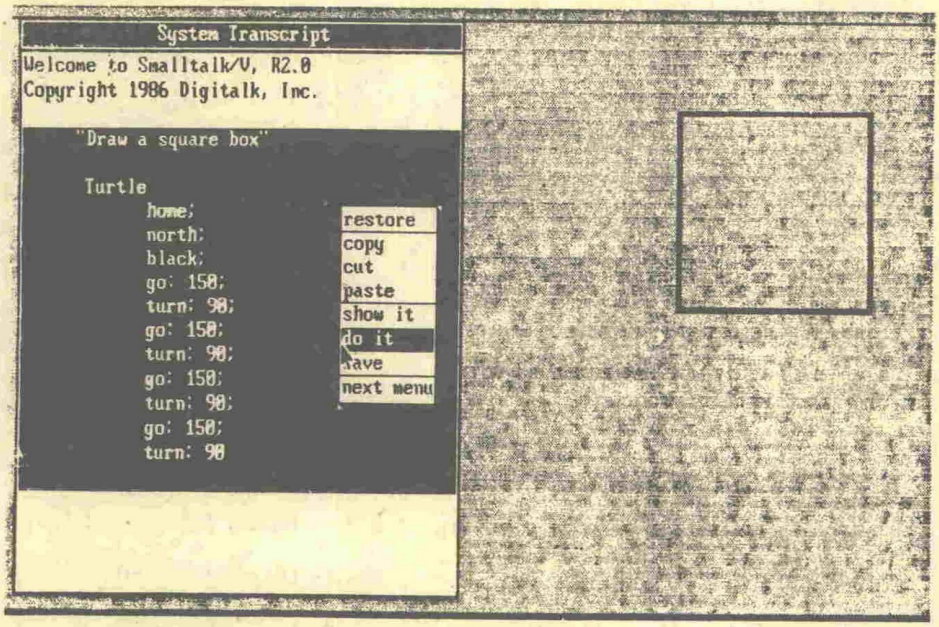
แล้วกด " + "

เริ่มทดลองเขียนโปรแกรมโดยเลือกเอาออปเจ็ค Turtle ซึ่งอยู่ใน Class ของ Pen ดูตามนี้คือจะมีปากกาด้ามหนึ่งในการวาดรูป ใน Class ของ Pen จะตอบสนองต่อ message หลาย ๆ อย่าง เช่น การกำหนดสี การกำหนดตำแหน่ง เป็นต้น ในที่นี้เราจะทำการวาดรูปสี่เหลี่ยมบนจอภาพ ให้ทำโดยพิมพ์ข้อความต่อไปนี้ลงบนหน้าต่างของ System Transcript

```
Turtle
    home;
    back:
    go:     150;
    turn:    90;
    go:     150;
    turn:    90;
    go:     150;
    turn:    90;
    go:     150;
    turn:    90
```



รูปที่ 3 การวาดรูป mandala



รูปที่ 4 ตัวอย่างอีกอันหนึ่งในการวาดรูป box สี่เหลี่ยม

จากนั้น ให้เลือกข้อความที่ให้ทำงานโดยเลื่อนเคอร์เซอร์ไปที่ท้ายสุดของข้อความ ถ้าเป็นเมาส์ให้กดปุ่มซ้ายสุดค้างเอาไว้ จากนั้นเลื่อนเคอร์เซอร์ไปที่อีกด้านหนึ่งของข้อความ (เป็นบล็อกสี่เหลี่ยม) ข้อความที่ถูกเลือกจะปรากฏเป็นลักษณะ reversed จากนั้นจึงค่อยปล่อยเมาส์

แต่ถ้าเป็นคีย์บอร์ด ให้เลื่อนเคอร์เซอร์ไปที่มุมใดมุมหนึ่งของข้อความแล้วกด " + " จากนั้นเลื่อนเคอร์เซอร์ต่อไปยังอีกด้านของข้อความแล้วกด " + " เช่นกัน ข้อความที่ถูกเลือกก็จะปรากฏเป็น reverse เช่นกัน

เมื่อเลือกข้อความได้แล้วให้กดปุ่ม DEL เพื่อเรียก Pane Menu ขึ้นมา เลือกหัวข้อ do it Smalltalk/V จะอ่านข้อความหรือโปรแกรมที่เราเลือกไว้มาทำงานโดยจะวาดรูปสี่เหลี่ยมบนจอภาพ

คราวนี้เราลองตัวอย่างอื่นต่อไป คือ

```
Turtle
    home;
    north;
    mandala: 12 diameter: 300
```

แล้วทำการเลือกข้อความหรือโปรแกรมตามวิธีที่กล่าวมาแล้ว และทำรัน (RUN) ซึ่งจะได้รูปของ mandala หนึ่งรูป เป็นส่วนหนึ่งของ multi mandala ในตัวอย่างที่ได้จาก demo ในรูปที่ 3

สรุป

Smalltalk/V มี class ให้งานมากหนึ่งพัน class ที่ตอบสนองต่อ message ถึง 2 หมื่นกว่า message ซึ่งสามารถนำมาใช้งานได้มากมาย นอกจากนี้ยังมีอีกหลายส่วนที่ไม่ได้กล่าวถึงในที่นี้ เช่น การกำหนด class ขึ้นมาใหม่ การสร้าง method ต่อการตอบสนองของ message ซึ่งจะขอยกเอาไว้ให้ผู้อ่านที่สนใจในเรื่องของ OPP ไปศึกษาเองจาก Smalltalk/V